Vers une meilleure modélisation du langage: la prise en compte des séquences dans les modèles statistiques

I. Zitouni, K. Smaïli

LORIA/INRIA-Lorraine
B.P.239 54506 Nancy, France
E-mail: {zitouni, smaili}@loria.fr

ABSTRACT

In natural language, several sequences of words are very frequent. Conventional language models do not adequately take into account such sequences, because they underestimate their probabilities. A better approach consists in modeling word sequences as if they were individual dictionary elements.
In this paper, we present an original method for automatically determining the most frequent phrases. This method is based on information theoretic criteria which insure a high statistical consistency, and on French grammatical classes which include additional linguistic dependencies. We propose also several language models based on theses word sequences. Experimental tests on a vocabulary of *20000* words show that the perplexity is reduced by more than *25%* compared to conventional models. The introduction of these word sequences in our dictation machine MAUD reduces the word error rate by more than *22%*.

1. INTRODUCTION

Une phrase est soumise à différentes contraintes : lexicales liées à la limitation du vocabulaire, syntaxiques et sémantiques régissant l'ordre des mots. Dès lors, toutes les combinaisons possibles de mots ne sont pas observables, loin s'en faut. Pour des raisons syntaxiques et/ou sémantiques, certaines suites de mots forment un groupe homogène. Ces séquences, naturellement de longueur variable, véhiculent souvent soit une idée soit une structure langagière particulière. Afin d'introduire cette notion importante dans les systèmes de reconnaissance automatique de la parole (RAP), nous avons généralisé plusieurs modèles de langage classiques en y incluant cette notion de séquences. Tout d'abord, nous avons considéré l'ensemble de ces séquences comme des unités de la langue. Ensuite, nous les avons ajoutées au vocabulaire de base, construisant ainsi un nouveau vocabulaire. Par conséquent, lors de la prédiction et même de la sélection, les modèles de langage utilisant ce nouveau vocabulaire se fondent sur un historique d'unités où chacune d'entre elles peut être, soit un mot, soit une séquence. Ceci donne aux modèles la possibilité d'utiliser un historique plus important et de mieux prendre en compte le rôle prédictif de ces séquences. Les modèles de langage correspondants sont également capables de prédire la totalité d'une séquence, et de ne plus se limiter à la prédiction d'un seul mot.
L'introduction de ces séquences peut augmenter le nombre d'unités (mots ou séquences de mots) peu fréquentes, ce qui risque de réduire les performances des modèles de langage. Les séquences ne doivent donc pas être introduites arbitrairement dans le vocabulaire initial. Nous présentons dans ce papier une nouvelle approche d'extraction de séquences de mots se fondant sur la perplexité et l'information mutuelle. Pour mieux prendre en compte les contraintes syntaxiques de la langue, nous utilisons également un ensemble de classes syntaxiques. A la différence des approches d'extraction des séquences, proposées par R-L. Mercer [1], E. Giachin [2], B.Suhm [3] et S. Deligne [4], celle-ci permet l'utilisation de grands vocabulaires sans nécessiter d'énorme capacité de calcul. Les résultats d'évaluation, présentés dans ce papier, sont estimés sur un vocabulaire de *20000* mots. Nous présentons les performances apportées par l'utilisation de ces séquences dans les nouveaux modèles de langage que nous proposons : n-SeqGrammes et n-SeqClasses correspondant respectivement à l'extension des n-grammes, et des n-classes. Pour mieux appréhender les contraintes langagières qui existent entre le mot courant et un historique lointain, nous proposons un modèle fondé sur la notion de triggers de séquences. Si une séquence A est fortement corrélée avec une séquence B, le couple (A-B) est considéré comme un trigger de séquence. Ainsi, si A apparaît dans l'historique, la vraisemblance de B est renforcé. L'originalité de cette approche, par rapport à celle couramment utilisée dans la littérature [5], est qu'elle se serve des séquences ; ce qui est linguistiquement plus riche. En effet, si on demande à une personne de prédire la suite de la phrase tronquée "le nom du président actuel est", il est plus naturel pour elle de répondre : "Jacques Chirac" plutôt que "Jacques".

2. INTERET DES SEQUENCES EN RAP

L'utilisation des séquences de mots, comme des unités à part entière dans un vocabulaire, permet aux modèles de langage de tenir compte de contraintes langagières supplémentaires ; ce qui accroîtra leurs performances. En effet, l'utilisation de ces séquences comporte de nombreux avantages :
L'historique traité par les modèles de langage à contexte fixe (n-grammes, n-classes, etc.) devient variable et plus long.
La prononciation orale d'une expression contenant plusieurs mots est souvent différente de celle où l'on prononce un à un (d'une façon indépendante) les mots qui la composent. Ainsi, le modèle à base de séquences apporte un plus au niveau acoustique par le biais de l'utilisation d'un modèle de prononciation spécifique à chaque séquence de mots.
Les résultats fournis par un système de RAP permettent

d'obtenir des phrases plus cohérentes au niveau linguistique que celles obtenues par des modèles à base de mots. Ceci est dû au fait que les séquences construites automatiquement ont souvent une structure linguistique viable.

3. ALGORITHME D'EXTRACTION DE SEQUENCES DE MOTS

L'algorithme d'extraction de séquences que nous proposons est entièrement automatique. Il se fonde sur un jeu de classes qui peut être déterminé soit manuellement soit automatiquement. Dans notre cas, nous utilisons un ensemble de classes syntaxiques V_c du français, construites manuellement par des experts ; un mot peut appartenir à plusieurs classes. L'algorithme commence ainsi par étiqueter automatiquement le corpus de mots *W*, avec l'ensemble des classes syntaxiques, construisant ainsi le corpus de classes *C*. Etiqueter le corpus revient à résoudre syntaxiquement le texte, autrement dit affecter à chaque mot sa classe syntaxique contextuelle. Nous utilisons pour ceci un algorithme de type Viterbi [6]. Ensuite, l'algorithme identifie dans *C* tous les couples de classes ou de séquences de classes adjacentes réduisant la perplexité. A partir de ces séquences de classes, on extrait les séquences de mots correspondant dans *W*. Ces unités ainsi construites constituent le résultat final de cet algorithme (les séquences de mots recherchées). Nous ne prenons en compte que les séquences de classes dont les mots correspondant appartiennent au vocabulaire *V*. Pour contrôler la convergence de l'algorithme, le nombre maximum de mots (*q*) dans une séquence est fixé *a priori*. Afin de limiter le nombre de séquences, l'algorithme prend en compte seulement les séquences de classes qui apparaissent fréquemment et qui ont une information mutuelle supérieure à un certain seuil T_J :

$$T_j = p \max_{c_i \in C, c_j \in C} J(c_i, c_j) \quad (1)$$

où *p* est un coefficient prédéfini et $J(c_i,c_j)$ représente l'information mutuelle du couple de classes ou de séquences de classes en argument :

$$J(c_i, c_j) = \log \frac{N(c_i, c_j) T}{N(c_i) N(c_j)} \quad (2)$$

où *N(.)* désigne l'occurrence de l'argument et *T* la taille du corpus. Une information mutuelle assez grande entre deux composants c_i et c_j indique que ces séquences ne sont pas apparues l'une à coté de l'autre, par un pur hasard, mais qu'elles constituent soit un groupe soit une partie d'un groupe au sens linguistique.

La prise en compte ou non d'une séquence de classes et d'une séquence de mots dépend du nombre minimal d'occurrences de chacune d'entre elles. Ces deux valeurs minimales sont notées T_{min} et T_{occ} dans l'algorithme de construction de séquences donné ci-dessous :

1) Déterminer, à partir du corpus C, les couples de classes ou de séquences de classes c_i,c_j ayant une information mutuelle supérieur à T_J et un nombre de constituants inférieur à *q*.

2) Ajouter ces séquences candidates de classes à V_c, construisant ainsi V_c'. Remplacer les par de nouvelles étiquettes dans *C*, formant ainsi *C'*.

3) A partir de ces séquences de classes, extraire les séquences de mots correspondantes dans le corpus *W*.

4) Ajouter les séquences $\{s_i,s_j\}$, obtenues dans la troisième étape, de fréquence supérieure à T_{occ} dans le vocabulaire *V*, construisant ainsi *V'*. Mettre à jour *W*, en replaçant ces séquences par de nouvelles étiquettes, formant ainsi *W'*.

5) Utiliser les vocabulaires *V'* et V_c' pour calculer la perplexité du modèle en utilisant le nouveau vocabulaire *V'*.

6) Si la perplexité diminue : affecter *V'* à *V*, *W'* à *W*, *C'* à *C* et retourner à la première étape.

7) Sinon, extraire des séquences ajoutées à *V'* celles qui réduisent la perplexité, une fois considéré comme unité.

Pour extraire les séquences de la septième étape, nous trions ces séquences en fonction de la valeur de l'information mutuelle des classes correspondantes et nous procédons par dichotomie sur chacune des deux moitiés. En effet, si la moitié en cours réduit la perplexité, nous l'additionnons à *V*. Sinon, nous la divisons en deux sous-ensembles, que nous traitons de nouveau.

La valeur de la perplexité est évaluée à chaque itération avant et après l'ajout des séquences candidates. Sur un corpus de *T* mots, cette valeur est calculée par un modèle biclasses comme suit :

$$PP = 2^{-\frac{1}{T}\log_2 P(w_1)\prod_{i=2}^{T} P(w_i / w_{i-1})} \quad (3)$$

où $P(w_i/w_{i-1})$ est définie par la formule suivante [7] :

$$P(w_i / w_{i-1}) = \sum_{c_i \in C_{w_i}} p(w_i / c_i) \times \sum_{c_{i-1} \in C_{w_{i-1}}} p(c_{i-1} / w_{i-1}) p(c_i / c_{i-1}) \quad (4)$$

où C_{w_i} est l'ensemble des classes syntaxiques auquel le mot w_i peut appartenir.

Pour estimer la valeur de la perplexité, il n'est pas nécessaire de parcourir, à chaque itération, toutes les unités (mots ou séquences de mots) du corpus. Il suffit de réestimer la vraisemblance des nouvelles séquences, de leurs unités voisines et d'extraire ainsi la nouvelle valeur de la perplexité [7]. Ce traitement améliore considérablement le temps de calcul nécessaire pour le déroulement de cet algorithme. Lorsque nous remplaçons les séquences de mots candidates par des unités, la taille du corpus se réduit et celle du vocabulaire s'agrandit. Pour pouvoir comparer les valeurs de perplexité, avant et après l'ajout des séquences candidates, nous gardons constant la valeur de *T* lors du calcul de la perplexité [8].

Pour extraire des séquences longues (exemple, *qu'elle heure est t'il*), plusieurs séquences courtes vont être générées (exemple, *qu'elle heure*). Certaines de ces séquences courtes ne sont plus utiles. Si le remplacement d'une séquence courte par ses constituant réduit la perplexité, celle-ci est supprimée.

Nous présentons dans la figure 1, la convergence en terme de perplexité de l'algorithme cité ci-dessus, en fonction du nombre d'itérations et de la valeur de *q* (le nombre maximum de mots dans une séquence). Les résultats montrent que cette procédure atteint son optimum pour une valeur de *q* égale à *6*, avec seulement *10* itérations.

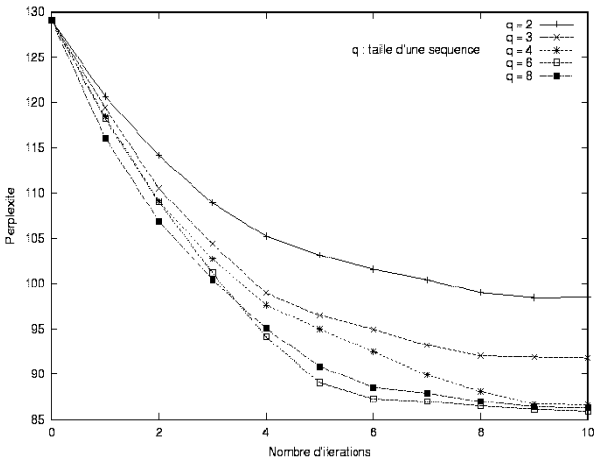

Figure 1 : convergence en terme de perplexité de l'algorithme d'extraction de séquences, à chaque itération, avec différentes valeurs de *q* (nombre maximum de mots dans une séquence).

4. MODELES A BASE DE SEQUENCES

Les modèles de langages couramment utilisés en reconnaissance automatique de la parole sont les n-grammes et les n-classes. En utilisant ces modèles, la vraisemblance d'un mot ne dépend que des *n-1* derniers mots de l'historique. Pour évaluer les performances des séquences, nous avons construit de nouveaux modèles n-grammes et n-classes utilisant le vocabulaire, résultat de l'algorithme présenté ci-dessus. Ce vocabulaire contient, en plus des mots du lexique de base, l'ensemble des séquences clés de mots ; ces séquences sont considérées comme des unités du vocabulaire. Nous notons par n-SeqGrammes et n-SeqClasses la généralisation des modèles n-grammes et n-classes respectivement, en y introduisant les séquences.

Pour prendre en compte les contraintes linguistiques existantes dans un historique distant, nous avons utilisé les modèles cache et triggers où l'unité de traitement est, soit un mot, soit une séquence de mots. Soit un trigger (M_1-M_2) ; dans la littérature, les unités M_1 et M_2 se limitent aux mots. Dans le modèle que nous proposons, elles peuvent être soit des mots soit des séquences de mots. Nous nommons SeqTriggers et SeqCache respectivement les modèles triggers et cache utilisant des séquences.

L'information mutuelle est une bonne mesure, permettant d'évaluer la quantité d'information apportée par un mot M_1 pour la prédiction d'un autre mot M_2. Nous avons ainsi utilisé cette mesure pour extraire les *K* meilleurs triggers (M_1-M_2) qui réduisent la perplexité [5]. Cette perplexité est calculée pour un modèle de langage, résultat d'une interpolation linéaire entre les modèles n-grammes, cache et triggers :

$$P(s/h) = \alpha P_{gram}(s/h) + \beta P_C(s/h) + \delta P_T(s/h) \quad (5)$$

où α, β et δ représentent les paramètres d'interpolation .

5. EVALUATION

5.1 Description des données

Le corpus utilisé (LeM) pour l'apprentissage des modèles de langage contient *43* millions de mots, représentant deux années (*87-88*) du journal « Le Monde ». Pour estimer les modèles n-classes et n-SeqClasses, nous avons utilisé un ensemble de *233* classes syntaxiques. Ces classes sont construites manuellement par des experts [6]. Le corpus de classes est celui obtenu par l'étiquetage de LeM. Le vocabulaire de base contient les *20000* mots les plus fréquents dans LeM.

5.2 Evaluation en terme de perplexité

Nous avons évalué nos modèles sur un corpus de test, contenant *5* million de mots qui n'ont pas servi à l'apprentissage. Signalons que l'application de l'algorithme d'extraction automatique de séquences nous a permis d'inclure 4000 nouvelles séquences dans le dictionnaire. Nous avons réparti nos modèles en deux catégories : la première (C_1) fondée sur l'utilisation exclusive des mots et la seconde (C_2) sur l'utilisation de séquences de longueur variable. La catégorie C_1 contient les modèles : bigrammes (P1), trigrammes (P2), biclasses (P3), triclasses (P4), un modèle obtenu par une interpolation linéaire entre bigrammes, cache et triggers (P5) et un modèle résultat d'une interpolation linéaire entre trigrammes, cache et triggers. La catégorie C_2 comporte les 2-SeqGrammes (PS1), 3-SeqGrammes (PS2), 2-SeqClasses (PS3), 3-SeqClasses (PS4), un modèle obtenu par une interpolation linéaire entre 2-SeqGrammes, SeqCache et un modèle résultat d'une interpolation linéaire entre 3-SeqGrammes, SeqCache et SeqTriggers. Pour apprendre l'ensemble de ces modèles, que se soit pour C_1 ou C_2, nous utilisons la méthode de Katz [9]. Nous exposons dans la table 1, les valeurs de perplexité de ces modèles de langage.

C_1	**P1**	**P2**	**P3**	**P4**	**P5**	**P6**
PP	*121.53*	*74.65*	*129.14*	*81.94*	*117.53*	*72.69*
C_2	**PS1**	**PS2**	**PS3**	**PS4**	**PS5**	**PS6**
PP	*83.63*	*63.96*	*85.87*	*72.12*	*80.00*	*60.95*

Table 1 : Comparaison en terme de perplexité des modèles de langage avec et sans séquences.

Une comparaison, en terme de perplexité entre l'ensemble de ces valeurs, montre que l'introduction des séquences améliore d'environ *25%* les performances des modèles de langage classiques.

5.3 Le système MAUD

L'amélioration des performances d'un modèle de langage en terme de perplexité ne suffit pas pour affirmer la supériorité d'un modèle par rapport à un autre, il faut

absolument le tester dans un cadre de reconnaissance. Pour ce faire ,nous avons intégré les modèles de langage que nous avons développé dans notre machine à dicter MAUD [10]. La version de base de MAUD se fonde sur un modèle probabiliste de langage et procède en *3* étapes : identification du genre du locuteur, construction d'un treillis de mots à partir d'un modèle acoustique dépendant du sexe et extraction de la meilleure hypothèse. Les deux premières passes utilisent un algorithme de type Viterbi-Bloc avec un modèle de langage bigrammes. La troisième étape se fonde sur un algorithme de recherche en faisceaux, sur un modèle trigramme et sur le score acoustique des mots reconnus lors de l'étape précédente.

Chaque phonème est représenté par un modèle de Markov de second ordre à trois états (HMM2) [11]. Un mot est ainsi défini par la concaténation des HMM2 des phonèmes qui le composent. Dans le cas d'une séquence, nous introduisons un HMM2 de silence optionnel entre les mots qui la composent. Pour estimer les HMM2 de phonèmes, nous utilisons le corpus de parole Bref80 [12].

Plusieurs versions du système ont été développées : **M1** représentant la version de base exposée ci-dessus ; **MS1** utilise un modèle 2-SeqGrammes lors de la deuxième passe et un modèle 3-SeqGrammes lors de la troisième passe ; **M2** est identique à M1 avec la différence qu'elle utilise un modèle biclasses et triclasses au lieu des modèles bigrammes et trigrammes respectivement ; **MS2** dans laquelle nous introduisons les séquences et nous remplaçons les modèles biclasses et triclasses de M2 par des modèles 2-SeqClasses et 3-SeqClasses respectivement ; **M3** ressemble à la version M1 et utilise un modèle résultat d'une interpolation linéaire entre les modèles trigrammes, cache et triggers ; **MS3** est identique à MS1 avec la différence qu'elle utilise en plus les modèles SeqTriggers et SeqCache lors de la troisième passe.

Nous présentons dans la table 2, le taux d'erreur de chacune des versions de MAUD citée ci-dessus. Nous estimons ce taux sur les 300 phrases de tests, fournies lors de la campagne de l'AUPELF-UREF d'évaluation des systèmes de reconnaissance automatique de la parole.

	M1	**MS1**	**M2**	**MS2**	**M3**	**MS3**
WER	*38%*	*29.9%*	*43%*	*33.5%*	*36.6%*	*28.1%*

Table 2 : Taux d'erreur (WER) des différentes versions du système MAUD avec et sans séquences de mots.

Les résultats de reconnaissance montrent que l'utilisation des séquences a permis de faire baisser d'environ *22%* le taux d'erreur de MAUD.

6. CONCLUSION ET PERSPECTIVES

Une nouvelle approche a été proposée dans ce travail, permettant d'améliorer les performances des modèles de langage et de celles de notre système de reconnaissance automatique de la parole. En utilisant cette approche, les modèles de langage sont évalués en utilisant un lexique de séquences de mots. Ces séquences sont extraites automatiquement, en se basant sur des critères d'optimalité connus en théorie de l'information : la perplexité et l'information mutuelle. Comparativement au modèle n'utilisant que les mots, le modèle à base de séquences a permis de réduire la valeur de la perplexité de *25%* et d'améliorer le taux de reconnaissance de MAUD d'environ *22%*. La majorité de nos séquences ont une structure syntaxique correcte. De plus, beaucoup d'entre elles sont de nature sémantique. L'originalité et la faisabilité de cette méthode sont dues au fait que le système générateur de séquences de mots est fondé sur des séquences de classes construites automatiquement. Ces séquences doivent elles-mêmes vérifier un certain nombre de contraintes (une information mutuelle supérieure à un certain seuil, un nombre limité de constituants et un contrôle permanent des séquences par le biais de la perplexité) avant d'être candidates à générer des séquences de mots.

Pour améliorer les performances de cette approche, nous comptons l'utiliser avec le modèle multigrammes [4]. Nous pensons également appliquer cette approche dans d'autres domaines : construction des classes d'équivalence sémantiques, identification thématique, traduction automatique, …

REFERENCES

[1] Jelinek F. "Self-Organized Language Modeling for Speech Recognition". In Readings in Speech Recognition, pp. 450-506. Ed. A. Waibel and K. F. Lee. Morgan Kaufmann, 1989.

[2] Giachin E. "Phrase Bigrams for Continuous Speech Recognition". ICASSP95, Detroit, pp. 225-228, 1995.

[3] Suhm B. and Waibel *A*. "Towards Better Language Models for Spontaneous Speech". ICSLP94, Yokohama, pp. 831-834, 1994.

[4] Deligne S. and Bimbot F. "Language Modeling by Variable Length Sequences: Theoretical Formulation and Évaluation of Multigrams". ICASSP95, Detroit, pp. 169-172, 1995.

[5] Tillmann C., Ney H., "Selection Criteria for Word Trigger Pairs in Language Modeling". In Grammatical Inference: Learning Syntax from Sentences, Springer, 1996.

[6] Smaïli K., Zitouni I., Charpillet F. and Haton J. P. "An Hybrid Language Model for a Continuous Dictation Prototype". Eurospeech97, Rhodes, pp. 2723-2726, 1997.

[7] Zitouni I. "Modlisation du langage pour les systèmes de reconnaissance de la parole destinés aux grands vocabulaire : application à MAUD". Thèse de l'université Henri Poincarré, Nancy, 2000.

[8] Adda G. et al., "Text Normalisation and Speech Recognition in French". Eurospeech97, Rhodes, 1997.

[9] Katz M. "Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer". IEEE Trans. ASSP, 35 (3), pp. 400-401, 1987.

[10] Fohr D, Haton J. P., Mari J. F., Smaïli K., Zitouni I. "MAUD: Un prototype de machine à dicter vocale". Actes 1ères JST FRANCIL, Avignon, pp. 25-30, 1997.

[11] Mari J. F., Haton J. P., Kriouile A. "Automatic Word Recognition Based on Second-Order Hidden Markov Models". IEEE Trans. ASSP, 2(1), pp. 22-25, 1997.

[12] Lamel L., Gauvain J-L., Eskenazi M. "BREF: a Large Vocabulary Spoken Corpus for French". Eurospeech91, Gênes, 1991.